REMARKS

In response to the Office Action mailed on March 27, 2008, Applicant respectfully requests reconsideration. Claims 1-28 and 30-31 are pending in this application. Claims 8-10, 12, 13, 22-24, 26, and 27 are objected to but would be allowable if rewritten in independent form. Claims 1-7, 11, 14-21, 25, 28 and 30 are rejected. Claim 29 is canceled. Claim 31 is new. Claims 1, 15, 29, and 30 are independent claims, and the remaining claims are dependent claims. Applicant thanks Examiner for indicating allowable subject matter. Applicant believes that the remaining claims as presented are in condition for allowance. A notice to this affect is respectfully requested.

Claim Amendments

Claim 31 is new. Claim 31 claims a process that includes all of the limitations of claim 1, and includes features from claims 2 and 11, and pages 7-8 from the specification. Thus no new matter has been added.

Rejections under 35 U.S.C. §102

Claims 1-7, 11, 14-21, 25, 28 and 30 were rejected under 35 USC 102 as being anticipated by Gerard et al. (US 6,023,704).

Applicant respectfully traverses these rejections. "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." MPEP 2131. Gerard fails to disclose or describe each and every element as set forth in the presently claimed invention.

General Discussion of Gerard.

Gerard relates to updating objects in an object oriented system. When changes to persistent objects are required, there is no uniform mechanism for updating the persistent objects. Gerard discloses a prior art method for updating

persistent objects, however such method requires that the system be shut down in order to passivate all the objects. The problem with this approach is that shutting down a large computer system for many hours is an unacceptable solution. Thus Gerard attempts to provide a mechanism to easily update persistent objects in an object oriented computer system without shutting down the computer system. The mechanism that Gerard provides dynamically updates persistent objects by swapping identities. Gerard takes a first object and uses the first object to instantiate a second object. Then Gerard swaps identities of the first and second objects and converts state data of the old object into the new object, thereby updating persistent objects without shutting down a system.

The present invention relates to a different problem. The present invention addresses situations for developing and executing software applications that rely on a common or shared object model. If both a client and server portions of a software application are developed using a common object model that is shared, and subsequent revisions to the application required changes to be made to that object model, then both the client and server software application code must be modified to accommodate those changes made to the shared object model. This is a problem because the client and server portions of a software application may be distributed in numerous computer systems worldwide. If a modification is made to an object model in a conventional development environment, it is not practical to update and test all instances of both server and client software application code to be compatible with one another. The present invention provides a solution without having to maintain two or more versions of an object model to provide backwards compatibility. The solution only requires a single current object model and does not require old methods of data definitions to be maintained or deprecated. Thus the current object model can be kept clean of former data definitions and method invocations no longer in use. The benefit of the solution is that if a customer upgrades a server portion of the software application, that customer is not

required to upgrade the client portion of the software application at the same time.

In a nutshell, Gerard relates to dynamically updating persistent objects in an object oriented computer system, while the present invention relates to backwards compatibility of shared object models.

INDEPENDENT:

Claim 1. The office action states that Gerard teaches "receiving a former client request requiring access to a former object defined by a former object model" at column 3 lines 50-52, and at column 4, lines 5-7. The cited sections of Gerard, however, simply teach how a pure object oriented program operates. In column 3, Gerard explains that objects are pieces of code that provide one or more services when requested by the client. In column 4, Gerard explains the client object sends request messages to server objects to perform a desired function, and then the server object receives and interprets the message to decide what operations to perform. Thus Gerard is silent on teaching "receiving a former client request requiring access to a former object defined by a former object model." In fact, Gerard is completely silent on the terms "former client request," "former object," and "former object model." Gerard is even silent on the word “former” and its synonyms.

The office action states that Gerard teaches "mapping a former object required for access by the former client request to a corresponding current object existing within a current object model" at column 2, lines 54-58, and at column 7, lines 43-46. The relevant sections of Gerard disclose: “According to preferred embodiments of the present invention, an object identity swapper dynamically updates the configuration of an object by taking a first object, instantiating a new second object, swapping the identities of the first and second objects”; and “Step 510 of method 500 instantiates a second object 127 as an instance of the second class that defines the desired new object configuration. Step 510 of figure 5

creates second object 127 of figure 6." Gerard is silent on mapping former objects, and objects required for access by the former client. Gerard is generally silent on mapping processes, as Gerard focuses on a method of swapping identities. Gerard essentially takes a first object, then instantiates a new second object with updated configuration, and then swaps identities so that the new second object is used. Thus, Gerard is silent on this limitation.

The office action states that Gerard teaches "copying current object data within the current object of the current object model to former object data within an instantiation of the former object" at column 2, lines 58-60, at column 8, lines 3-13, and in figure 8. Gerard, however, fails to teach this limitation. In column 2, Gerard discloses "reading and converting the state data of the old object (now the second object) into the new object (now the first object). Because Gerard deals with swapping objects, it is important to convert the state data so that the first object functions in place of the second object. In column 8, Gerard details the process of how the state data is converted to be used in the swapped object. Thus Gerard is silent on copying current object data to former object data within an instantiation of the former object.

The office action states that Gerard teaches "processing the former client requests using the instantiation of the former object to satisfy the former client request" at column 4, lines 5-7. In column 4, Gerard explains the client object sends request messages to server objects to perform desired function, and then the server object receives and interprets the message to decide what operations to perform. Thus Gerard is silent on processing former client requests using an instantiation of a former object to satisfy the former client request.

Claim 15. Claim 15 claims a computer system having a memory, processor, a communications interface, and an interconnection mechanism. The memory is encoded with a server application that causes a computer system to perform operations including all the process steps of claim 1. Gerard discloses a memory and a processor, however, Gerard fails to disclose the process steps as

explained above. Therefore claim 15 is believed to be allowable under the same rationale as claim 1.

Claim 30. Claim 30 claims an article of manufacture, that is, computer program product having to computer-readable medium including computer program logic encoded thereon. When the computer program logic is executed on a computer system, the computer program product of claim 30 executes all the process steps of claim 1. Therefore claim 30 is believed to be allowable under the same rationale as claim 1.

DEPENDENT:

Claims 2-7, 11, 14, 16-21, 25 and 28 each depend on either claim 1 or claim 15. Because each of the dependent claims incorporates all of the limitations of either claim 1 or claim 15, the dependent claims are believed to be allowable for the reasons discussed above.

Summary

The claims are not anticipated by Gerard because Gerard fails to disclose all claim features. Thus the claims are believed to be in condition for allowance.

Applicant hereby petitions for any extension of time which is required to maintain the pendency of this case. If there is a fee occasioned by this response, including an extension fee, that is not covered by an enclosed check, please charge any deficiency to Deposit Account No. 50-3735.

Respectfully submitted,

/DWR/

David D. Rouille, Esq.
Attorney for Applicant(s)
Registration No.: 40,150
Chapin Intellectual Property Law, LLC
Westborough Office Park
1700 West Park Drive, Suite 280
Westborough, Massachusetts 01581
Telephone: (508) 616-9660
Facsimile: (508) 616-9661

Attorney Docket No.: EMC04-05(04034)

Dated: June 27, 2008